**STL 과제 보고서**

과목명: STL  
학번: 2020182048  
이름: 김정학

**과제를 해결하기 위해 고민한 내용과 방법**

일단 과제pdf 파일을 열어보니 교수님께서 Player 2백만개의 정보를 binary 모드로 기록하였다고 작성하셨다. Player 클래스의 정의만으로는 파일이 실제로 어떻게 쓰여지는지 잘 모르겠어, 비주얼 스튜디오로 “2022 STL 과제 파일”을 열어보았다. 파일에는 binary로 player의 정보가 쓰여 있었고, Player 객체의 num만큼의 바이트에 해당하는 정보가 쓰여 있는 것을 확인했다. 따라서 파일에 쓰여진 플레이어를 차례대로 읽기 위해서는 player의 정보를 읽고 그 player의 num만큼의 바이트를 읽어야 한다는 것을 알아냈다.

이 생각을 실제 코드로 옮겨 정말로 200만개만큼 읽어지는지 테스트를 해보았는데, 이상하게 잘 읽어져서 중단점을 걸고 비주얼 스튜디오 디버깅 툴을 사용해 읽어온 데이터를 확인해 보았는데, 아무것도 읽히지 않는 것을 확인했다. 순간 떠오른 생각으로, 파일이 정상적으로 열렸는지 확인을 해봐야 한다는 것이었다. 파일이 제대로 열렸는지 확인해보았더니, 아니나 다를까 제대로 열리지 않았다는 것이 판명이 되었고 만들어진 “2022 STL 과제 파일”이 소스 파일과 같은 폴더 내에 있지 않았다. 과제 파일을 소스 파일과 동일한 경로에 두고 프로그램을 돌리자마자 boundary exception이 발생했다. Player의 num만큼 바이트를 읽어오는데, 그냥 char\*에는 num만큼의 바이트를 저장할 수가 없기 때문이었다. 이를 위해서 num만큼의 바이트를 동적할당을 하고 player의 p에 할당된 메모리의 시작주소를 저장하는 방법으로 파일을 계속 읽었다. 결과적으로 파일을 끝까지 읽는데 성공하였고, 실제로 파일에는 2백만개의 player의 정보들이 적혀 있음을 알 수 있었다. 처음에는 STL 컨테이너인 vector를 사용하여 데이터를 읽어냈지만, 교수님의 player 객체는 추가되거나 삭제되지 않는다는 말씀이 array를 사용해도 된다는 것임을 나중에 알게 된 후에, vector를 array로 대체하였다. 파일의 끝까지 정상적으로 읽는지 확인하기 위해서 while 반복문의 조건식으로 파일 스트림으로 연 파일을 그냥 넣어서 끝인지 아닌지 판별을 했는데, 이런 방법으로 체크를 하면 읽어 나가는 과정에서 마지막 객체를 읽고 난 후, 파일의 끝(end of file)에 도달하는 것이 아닌, 파일의 마지막 끝 부분을 가리키게 되어서, 마지막 객체를 한 번 더 읽는 버그가 발생하게 된다. 이를 해결하기 위해서, 처음에는 vector의 pop을 사용해서 마지막 원소를 지우는 방법을 선택했는데, 아무리 생각해도 이 방법은 이상해서 교수님께 메일을 통해 질문을 했는데, 파일 스트림의 read 함수를 조건으로 체크를 할 수 있다고 수업시간에 배운 내용을 상기시켜 주셨다. 결론적으로, Player 200만개를 파일에서 잘 읽어왔다. 파일을 읽은 후에는 대부분이 STL 알고리즘 함수들과 컨테이너 내장 함수 등으로 해결이 가능한 부분으로 생각했다. 문제 1번에는 array 컨테이너의 마지막 원소를 반환하는 내장 함수 back을 사용하고 player 클래스에 선언한 show 함수를 사용하여 그 객체의 정보를 출력하는 방법을 사용하여 해결했다. 문제 2번에서는 array에 담겨있는 모든 Player들의 점수를 표준 템플릿 라이브러리인 numeric에 있는 accumulate함수를 사용하여 합계를 long long int에 저장해서 나누는 식으로 했는데, 다른 사람들과 의견을 나누어 보니 총합을 객체의 개수로 나누는 과정에서 소수점 아래 자리의 숫자가 생길 수도 있다고 생각이 들었다. 따라서 플레이어들의 점수 총합을 long long int 자료형에 담고, 나누는 과정에서 double로 형 변환을 한 후 객체 개수로 나누어서 출력을 하였다.  
텍스트이(가) 표시된 사진

자동 생성된 설명

문제3번의 경우에는 조건에 맞는 객체의 수를 알아내기 위해 STL 알고리즘 함수 중 하나인 count\_if를 사용하여 조건에 맞는, 즉 num이 999인 player들의 개수를 세고, 교수님이 작성하신 player 클래스 함수인 write로 “바이트 수 999”라는 파일에 binary 모드로 작성하였다. 해당 조건에 맞는 player의 개수는 2024개이며, 작성된 파일의 크기는 2,135,320바이트, 디스크에 할당된 크기는 2,138,112라고 파일의 속성에는 나와있다. 혹시나 하는 마음에 “바이트 수 999”파일을 읽어서 개수를 확인해보았더니, 2023개로 출력이 되었다.